Fast Discovery of Representative Association Rules

Marzena Kryszkiewicz

Institute of Computer Science, Warsaw University of Technology
Nowowiejska 15/19, 00-665 Warsaw, Poland
mkr@ii.pw.edu.pl

Abstract. Discovering association rules among items in a large database is an important database mining problem. The number of association rules may be huge. To alleviate this problem, we introduced in [1] a notion of representative association rules. Representative association rules are a least set of rules that covers all association rules satisfying certain user specified constraints. The association rules, which are not representative ones, may be generated by means of a cover operator without accessing a database. In this paper, we investigate properties of representative association rules and offer a new efficient algorithm computing such rules.

1 Introduction

Discovering association rules among items in large databases is recognized as an important database mining problem. The problem was introduced in [2] for sales transaction database. The association rules identify sets of items that are purchased together with other sets of items. For example, an association rule may state that 90% of customers who buy butter and bread buy also milk. Several extensions of the notion of an association rule were offered in the literature (see e.g. [3-4]). One of such extensions is a generalized rule that can be discovered from a taxonomic database [3]. Applications for association rules range from decision support to telecommunications alarm diagnosis and prediction [5-6].

The number of association rules is usually huge. A user should not be presented with all of them, but rather with these which are original, novel, interesting. There were proposed several definitions of what is an interesting association rule (see e.g. [3,7]). In particular, pruning out uninteresting rules which exploits the information in taxonomies seems to be quite useful (resulting in the rule number reduction amounting to 60% [3]). The interestingness of a rule is usually expressed by some quantitative measure. In [1] we offered a different approach. We did not introduce any measure defining interestingness of a rule, but we showed how to derive the set of association rules from a given association rule by means of a cover operator without accessing a database. A least set of association rules that allows to deduce all other rules satisfying user specified constraints is called a set of representative association rules. In [1], it was offered the *GenAllRepresentatives* algorithm computing representative

L. Polkowski and A. Skowron (Eds.): RSCTC'98, LNAI 1424, pp. 214-222, 1998.

association rules. To check whether a candidate rule is representative the algorithm required comparing the rule with longer representative rules, which was quite time-consuming operation. In this paper, we investigate some properties of representative association rules that allow us to propose a new efficient algorithm for representative association rules mining. The new algorithm generates representative rules independently from other representative rules.

2 Association Rules

The definition of a class of regularities called *association rules* and the problem of their discovering were introduced in [2]. Here, we describe this problem after [2,8]. Let $I = \{i_1, i_2, ..., i_m\}$ be a set of distinct literals, called *items*. In general, any set of items is called an *itemset*. Let D be a set of transactions, where each transaction T is a set of items such that $T \subseteq I$. An *association rule* is an expression of the form $X \Rightarrow Y$, where $\emptyset \neq X,Y \subset I$ and $X \cap Y = \emptyset$. X is called the antecedent and Y is called the consequent of the rule.

Statistical significance of an itemset X is called *support* and is denoted by $sup(X)$. $Sup(X)$ is defined as the number of transactions in D that contain X. Statistical significance (*support*) of a rule $X \Rightarrow Y$ is denoted by $sup(X \Rightarrow Y)$ and defined as $sup(X \cup Y)$. Additionally, an association rule is characterized by *confidence*, which expresses its strength. The confidence of an association rule $X \Rightarrow Y$ is denoted by $conf(X \Rightarrow Y)$ and defined as the ratio $sup(X \cup Y) / sup(X)$.

The problem of mining association rules is to generate all rules that have support greater than some user specified minimum support $s \geq 0$ and confidence not less than a user specified minimum confidence $c > 0$. In the sequel, the set of all association rules whose support is greater than s and confidence is not less than c will be denoted by $AR(s,c)$. If s and c are understood then $AR(s,c)$ will be denoted by AR.

In the paper, we apply also the following simple notions:

The number of items in an itemset will be called the *length of the itemset*. An itemset of the length k will be referred to as a *k-itemset*. Similarly, the *length of an association rule* $X \Rightarrow Y$ will be defined as the total number of items in the rule's antecedent and consequent ($| X \cup Y |$). An association rule of the length k will be referred to as a *k-rule*. An association k-rule will be called *shorter* than, *longer* than or *of the same length* as an association m-rule if $k < m$, $k > m$, or $k = m$, respectively.

3 Cover Operator

A notion of a *cover operator* was introduced in [1] for deriving a set of association rules from a given association rule without accessing a database.

The *cover* C of the rule $X \Rightarrow Y$, $Y \neq \emptyset$, is defined as follows:

$C(X \Rightarrow Y) = \{X \cup Z \Rightarrow V |\ Z,V \subseteq Y \text{ and } Z \cap V = \emptyset \text{ and } V \neq \emptyset\}$.

Each rule in $C(X \Rightarrow Y)$ consists of a subset of items occurring in the rule $X \Rightarrow Y$. The antecedent of any rule r covered by $X \Rightarrow Y$ contains X and perhaps some items from Y, whereas r's consequent is a non-empty subset of the remaining items in Y. It was proved in [1] that each rule r in the cover $C(r')$, where r' is an association rule having support s and confidence c, belongs in $AR(s,c)$. Hence, if r belongs in $AR(s,c)$ then every rule r' in $C(r)$ also belongs in $AR(s,c)$. The number of different rules in the cover of the association rule $X \Rightarrow Y$ is equal to $3^m - 2^m$, where $m = |Y|$ (see [1]).

Example 3.1
Let $T_1 = \{A,B,C,D,E\}$, $T_2 = \{A,B,C,D,E,F\}$, $T_3 = \{A,B,C,D,E,H,I\}$, $T_4 = \{A,B,E\}$ and $T_5 = \{B,C,D,E,H,I\}$ are the only transactions in the database D. Let r: $(AB \Rightarrow CDE)$. Fig. 1 contains all rules belonging in the cover $C(r)$ along with their support and confidence in D. The support of r is equal to 3 and its confidence is equal to 75%. The support and confidence of all other rules in $C(r)$ are not less than the support and confidence of r.

#	Rule r' in $C(r)$	Support of r'	Confidence of r'
1.	$AB \Rightarrow CDE$	3	75%
2.	$AB \Rightarrow CD$	3	75%
3.	$AB \Rightarrow CE$	3	75%
4.	$AB \Rightarrow DE$	3	75%
5.	$AB \Rightarrow C$	3	75%
6.	$AB \Rightarrow D$	3	75%
7.	$AB \Rightarrow E$	4	100%
8.	$ABC \Rightarrow DE$	3	100%
9.	$ABC \Rightarrow D$	3	100%
19.	$ABC \Rightarrow E$	3	100%
11.	$ABD \Rightarrow CE$	3	100%
12.	$ABD \Rightarrow C$	3	100%
13.	$ABD \Rightarrow E$	3	100%
14.	$ABE \Rightarrow CD$	3	75%
15.	$ABE \Rightarrow C$	3	75%
16.	$ABE \Rightarrow D$	3	75%
17.	$ABCD \Rightarrow E$	3	100%
18.	$ABCE \Rightarrow D$	3	100%
19.	$ABDE \Rightarrow C$	3	100%

Fig. 1. The cover of the association rule r: $(AB \Rightarrow CDE)$

Below, we present two simple properties, which will be used further in the paper.

Property 3.1
Let r: $(X \Rightarrow Y)$ and r': $(X' \Rightarrow Y')$ be association rules. Then:

$$r \in C(r') \text{ iff } X \cup Y \subseteq X' \cup Y' \text{ and } X \supseteq X'.$$

Property 3.2
(i) If an association rule r is longer than an association rule r' then $r \notin C(r')$.

(ii) If an association rule r: $(X \Rightarrow Y)$ is shorter than an association rule r': $(X' \Rightarrow Y')$ then $r \in C(r')$ iff $X \cup Y \subset X' \cup Y'$ and $X \supseteq X'$.

(iii) If r: ($X \Rightarrow Y$) and r': ($X' \Rightarrow Y'$) are different association rules of the same length then $r \in C(r')$ iff $X \cup Y = X' \cup Y'$ and $X \supset X'$.

4 Representative Association Rules

In this section we describe a notion of representative association rules which was introduced in [1]. Informally speaking, a set of all representative association rules is a least set of rules that covers all association rules by means of the cover operator.

A set of *representative association rules* wrt. minimum support s and minimum confidence c will be denoted by $RR(s,c)$ and defined as follows:

$$RR(s,c) = \{r \in AR(s,c) | \neg \exists r' \in AR(s,c),\ r' \neq r \text{ and } r \in C(r')\}.$$

If s and c are understood than $RR(s,c)$ will be denoted by RR. Each rule in RR is called a *representative association rule*. By the definition of RR no representative association rule may belong in the cover of another association rule.

Example 4.1
Given minimum support s = 3 and minimum confidence c = 75%, the following representative rules $RR(s,c)$ would be found for the database D from Example 3.1:

$$\{A \Rightarrow BCDE,\ C \Rightarrow ABDE,\ D \Rightarrow ABCE,\ B \Rightarrow CDE,\ E \Rightarrow BCD,\ B \Rightarrow AE,\ E \Rightarrow AB\}.$$

There are 7 representative association rules in $RR(s,c)$, whereas the number of all association rules in $AR(s,c)$ is 165. Hence, the representative association rules constitute 4.24% of all association rules.

We may expect that a user will often request the set of representative association rules RR rather than the set of all association rules AR. If RR is provided then the user may formulate queries about the association rules represented by RR. Clearly, $AR(s,c) = \bigcup \{C(r) | r \in RR(s,c)\}$. However, we expect the user to ask rather about the covers of specific representative rules. The queries might contain not only the cover operator, but also the set-theoretical operators of union, difference and intersection.

5 The Algorithm

The problem of generating association rules is usually decomposed into two subproblems:

1. Generate all itemsets whose support exceeds the minimum support s. The itemsets of this property are called *frequent* (*large*).
2. From each frequent itemset generate association rules whose confidence is not less than the minimum confidence c. Let Z be a frequent itemset and $\emptyset \neq X \subset Z$. Then any rule $X \Rightarrow Z \backslash X$ holds if $sup(Z)/sup(X) \geq c$.

In the paper we restrict the second subproblem to generation of all representative association rules whose confidence is not less than the minimum confidence *c*.

Several efficient solutions were proposed to solve the first subproblem (see [3,8-9]). We will remind briefly the main idea of the *Apriori* algorithm [8] computing frequent itemsets. Then, we will propose a new efficient algorithm computing representative association rules from the found frequent itemsets.

5.1 Computing Frequent Itemsets

The *Apriori* algorithm exploits the following properties of frequent and non-frequent itemsets: All subsets of a frequent itemset are frequent and all supersets of a non-frequent itemset are non-frequent. The following notation is used in the *Apriori* algorithm:

- C_k - set of candidate *k*-itemsets;
- F_k - set of frequent *k*-itemsets;

The items in itemsets are assumed to be ordered lexicographically. Associated with each itemset is a *count* field to store the support for this itemset.

```
Algorithm Apriori
 F1 = {frequent 1-itemsets};
 for (k = 2; Fk-1 ≠ ∅; k++) do begin
  Ck = AprioriGen(Fk-1);
  forall transactions T ∈ D do
   forall candidates Z ∈ Ck do
    if Z ⊆ T then
     Z.count++;
  Fk = {Z ∈ Ck | Z.count > s};
 endfor;
return ∪k Fk;
```

First, the support of all 1-itemsets is determined during one pass over the database D. All non-frequent 1-itemsets are discarded. Then the loop "for" starts. In general, some *k*-th iteration of the loop consists of the following operations:

1. *AprioriGen* is called to generate the candidate *k*-itemsets C_k from the frequent (*k*-1)-itemsets F_{k-1}.
2. Supports for the candidate *k*-itemsets are determined by a pass over the database.
3. The candidate *k*-itemsets that do not exceed the minimum support are discarded; the remaining *k*-itemsets F_k are found frequent.

```
function AprioriGen(frequent (k-1)-itemsets Fk-1);
 insert into Ck
  select (Z[1], Z[2], ... , Z[k-1], Y[k-1]) from Fk-1 Z, Fk-1 Y
  where Z[1] = Y[1] ∧ ... ∧ Z[k-2] = Y[k-2] ∧ Z[k-1] < Y[k-1];
 delete all itemsets Z ∈ Ck such that some (k-1)-subset of Z
  is not in Fk-1;
return Ck;
```

The *AprioriGen* function constructs candidate *k*-itemsets as supersets of frequent (*k*-1)-itemsets. This restriction of extending only frequent (*k*-1)-itemsets is justified

since any k-itemset, which would be created as a result of extending a non-frequent (k-1)-itemset, would not be frequent either. The last operation in the *AprioriGen* function prunes the candidates from C_k that do not have all their (k-1)-subsets in the frequent (k-1)-itemsets F_{k-1}. If k-itemset Z does not have all its (k-1)-subsets in F_{k-1} then there is some non-frequent (k-1)-itemset $Y \notin F_{k-1}$ which is a subset of Z. This means that Z is non-frequent as a superset of a non-frequent itemset.

5.2 Computing Representative Association Rules

In this subsection we offer an efficient algorithm for computing representative association rules. Unlike the *GenAllRepresentatives* algorithm proposed in [1], the new *FastGenAllRepresentatives* algorithm exploits solely the information about the supports of frequent itemsets. The *Apriori* algorithm may be run to calculate all frequent itemsets and their supports. *FastGenAllRepresentatives* is based on Properties 5.2.1-5.2.2, which we present and prove below.

Lemma 5.2.1
Let $\emptyset \neq X \subset Z \subseteq I$ and r be an expression of the form $(X \Rightarrow Z \backslash X)$.

$$\exists Z' \subseteq I,\ Z' \supset Z \text{ and } sup(Z') > s \text{ and } sup(Z')/sup(X) \geq c \text{ iff}$$

$$\exists r' \in AR(s,c),\ r' \text{ is longer than } r \text{ and } r \in C(r').$$

Proof:

($\Rightarrow$) Let r' be an expression of the form: $X \Rightarrow Z' \backslash X$. Clearly, r' is longer than r since $Z' \supset Z$. The rule $r' \in AR(s,c)$ because $X \neq \emptyset$, the support $sup(r') = sup(Z') > s$ and the confidence $conf(r') = sup(Z')/sup(X) \geq c$. Additionally, Property 3.1 allow us to conclude that $r \in C(r')$.

($\Leftarrow$) Let $Z' \subseteq I$ and r' be an expression of the form: $X \Rightarrow Z' \backslash X$. By the assumption, $r' \in AR(s,c)$. Hence, $sup(r') = sup(Z') > s$ and $conf(r') = sup(Z')/sup(X) \geq c$. Additionally, r' is longer than r and $r \in C(r')$. Therefore, we can conclude from Property 3.2.ii that $Z' \supset Z$.

Lemma 5.2.2
Let $\emptyset \neq X \subset Z \subseteq I$ and r be an expression of the form $(X \Rightarrow Z \backslash X)$. Let $maxSup = max(\{sup(Z') |\ Z \subset Z' \subseteq I\} \cup \{0\})$.

$maxSup > s$ and $maxSup/sup(X) \geq c$ iff $\exists r' \in AR(s,c)$, r' is longer than r and $r \in C(r')$.

Proof: Lemma 5.2.2 follows immediately from Lemma 5.2.1.

Property 5.2.1
Let $\emptyset \neq X \subset Z \subseteq I$ and r be a rule: $(X \Rightarrow Z \backslash X) \in AR(s,c)$. The rule r belongs in $RR(s,c)$ if the two following conditions are satisfied:

(i) $maxSup \leq s$ or $maxSup/sup(X) < c$, where
$maxSup = max(\{sup(Z') |\ Z \subset Z' \subseteq I\} \cup \{0\})$,

(ii) $\neg\exists X', \emptyset \neq X' \subset X$, such that $(X' \Rightarrow Z \backslash X') \in AR(s,c)$.

Proof: Property 3.2.i tells us that an association rule does not belong in the cover of any shorter rule. So, the association rule r is representative if it does not belong in the cover of any association rule different from r which is longer than r or which is of the same length as r. The first condition (i) guarantees that the rule r does not belong in the cover of any association rule longer than r (see Lemma 5.2.2). The second condition (ii) ensures that the rule r does not belong in the cover of any association rule of the same length as r (see Property 3.2.iii).

Property 5.2.2

Let $\emptyset \neq Z \subset Z' \subseteq I$. If $sup(Z) = sup(Z')$ then no rule $(X \Rightarrow Z \backslash X) \in AR(s,c)$, where $\emptyset \neq X \subset Z$, belongs in $RR(s,c)$.

Proof: Let $(X \Rightarrow Z \backslash X) \in AR(s,c)$. Then, $\emptyset \neq X$, $sup(Z) > s$ and $conf(X \Rightarrow Z \backslash X) \geq c$. Now, let us consider a rule: $X \Rightarrow Z' \backslash X$. $(X \Rightarrow Z' \backslash X) \in AR(s,c)$ because $\emptyset \neq X$, $sup(Z') = sup(Z) > s$ and $conf(X \Rightarrow Z' \backslash X) = sup(Z')/sup(X) = conf(X \Rightarrow Z \backslash X) \geq c$. Additionally, $(X \Rightarrow Z \backslash X) \in C(X \Rightarrow Z' \backslash X)$. Hence, $X \Rightarrow Z \backslash X$ is not representative.

```
procedure FastGenAllRepresentatives(all frequent itemsets F);
 forall Z ∈ F do begin
  k = |Z|; maxSup = max({sup(Z')| Z⊂Z'∈F_{k+1}} ∪ {0});
  if Z.sup ≠ maxSup then begin            // see Property 5.2.2
   A_1 = {{Z[1]}, {Z[2]}, ... , {Z[k]}}; // create 1-antecedents
   /* Loop1 */
   for (i = 1; (A_i ≠ ∅) and (i < k); i++) do begin
    forall X ∈ A_i do begin
     find Y∈F_i such that Y = X;
     XCount = Y.count;
     /* Is X ⇒ Z\X an association rule? */
     if (Z.count/XCount ≥ c) then begin
      /* Aren't there representatives longer than X ⇒ Z\X? */
      if (maxSup/XCount < c) then    // see Property 5.2.1.i
       print(X, "⇒", Z\X, " with support: ", Z.count,
             " and confidence: ", Z.count / XCount);
      /* Antecedents of association rules are not extended */
      A_i = A_i \ {X};                  // see Property 5.2.1.ii
     endif;
    endfor;
    A_{i+1} = AprioriGen(A_i);           // compute i+1-antecedents
   endfor;
  endif;
 endfor;
endproc;
```

The *FastGenAllRepresentatives* algorithm computes representative association rules from each itemset in F. Let Z be a considered itemset in F. Only k-rules, $k = |Z|$, are generated from Z. First, *maxSup* is determined as a maximum from the supports of these itemsets in F_{k+1} which are supersets of Z. If there is no superset of Z in F_{k+1} then *maxSup*=0. Let us note that the supports of other proper supersets of Z, which do not belong in F_{k+1}, are not greater than *maxSup*. Clearly, *maxSup*>s or *maxSup*=0. If $sup(Z)$ is the same as *maxSup* then no representative rule can be generated from Z (see

Property 5.2.2). Otherwise, single-item antecedents of candidate *k*-rules are created. Loop1 starts. In general, the *i*-th iteration of Loop1 looks as follows:

Each candidate $X \Rightarrow Z \backslash X$, where $X \subset Z$ belongs in *i*-itemsets A_i, is considered. Z is frequent, so X, which is a subset of Z, is also frequent. In order to check if $X \Rightarrow Z \backslash X$ is an association rule its confidence: $sup(Z)/sup(X)$ has to be determined. $sup(Z)=Z.count$, while $sup(X)$ is computed as $sup(Y)$ of a frequent itemset Y in F_i such that $Y=X$. Only association rules that satisfy both conditions of Property 5.2.1 are representative. Condition (ii) is satisfied for any antecedent X which is 1-itemset. Proper generating of antecedents makes this condition true also for consequent sets A_i. So, in order to state whether an association rule is representative it is enough to check if condition (i) of Property 5.2.1 holds, i.e. whether $maxSup \leq s$ or $maxSup/sup(X)<c$. If $maxSup=0$ then both subconditions are satisfied, so $X \Rightarrow Z \backslash X$ is representative. Otherwise $maxSup>s$, which means that the letter subcondition will decide if $X \Rightarrow Z \backslash X$ is representative. The antecedent X of each association rule $X \Rightarrow Z \backslash X$ is removed from A_i. Having found all representative *k*-rules with *i*-antecedents from Z, $(i+1)$-itemset antecedents A_{i+1} are built from A_i by the *AprioriGen* function. In the result A_{i+1}, does not contain any itemset X such that $X \Rightarrow Z \backslash X$ would belong in the cover of another association rule $X' \Rightarrow Z \backslash X'$ such that $X' \subset X$. Therefore statement (ii) of Property 5.2.1 is an invariant of the algorithm.

6 Conclusions

In the paper we investigated properties of association rules that allowed us to construct an efficient algorithm computing representative association rules. Unlike the algorithm proposed in [1], the new algorithm exploits solely the information about the supports of frequent itemsets.

References

1. Kryszkiewicz, M.: Representative Association Rules. In: Proc. of PAKDD '98. Melbourne, Australia. Lecture Notes in Artificial Intelligence. Springer-Verlag (1998)
2. Agraval, R., Imielinski, T., Swami, A.: Mining Associations Rules between Sets of Items in Large Databases. In: Proc. of the ACM SIGMOD Conference on Management of Data. Washington, D.C. (1993) 207-216
3. Srikant, R., Agraval, R.: Mining Generalized Association Rules. In: Proc. of the 21st VLDB Conference. Zurich, Swizerland (1995) 407-419
4. Meo, R., Psaila, G., Ceri, S.: A New SQL-like Operator for Mining Association Rules. In: Proc. of the 22nd VLDB Conference. Mumbai (Bombay), India (1996)
5. Communications of the ACM, November 1996, Vol. 39. No 11. (1996)
6. Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (eds.): Advances in Knowledge Discovery and Data Mining. AAAI, Menlo Park, California (1996)

7. Piatetsky-Shapiro, G.: Discovery, Analysis and Presentation of Strong Rules. In: Piatetsky-Shapiro, G., Frawley, W. (eds.): Knowledge Discovery in Databases. AAAI/MIT Press, Menlo Park, CA (1991) 229-248
8. Agraval, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.I.: Fast Discovery of Association Rules. In: [6] (1996) 307-328
9. Savasere, A, Omiecinski, E., Navathe, S.: An Efficient Algorithm for Mining Association Rules in Large Databases. In: Proc. of the 21st VLDB Conference. Zurich, Swizerland (1995) 432-444